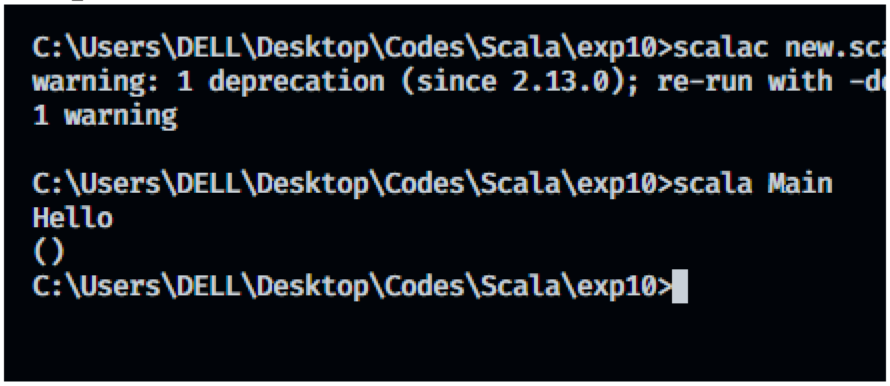
<u>SAPID : 60009200030</u> – Sarvagya Singh – K1
<u>Scala Experiment – lab 7</u>

1.
Code :

```
object Main {
   def main(args: Array[String]) {
        var result = search ("Hello")
        print(result)
    }
    def search (a:Any):Any = a match{
        case 1  => println("One")
        case "Two" => println("Two")
        case "Hello" => println("Hello")
        case _ => println("No")

        }
}
```

Output:

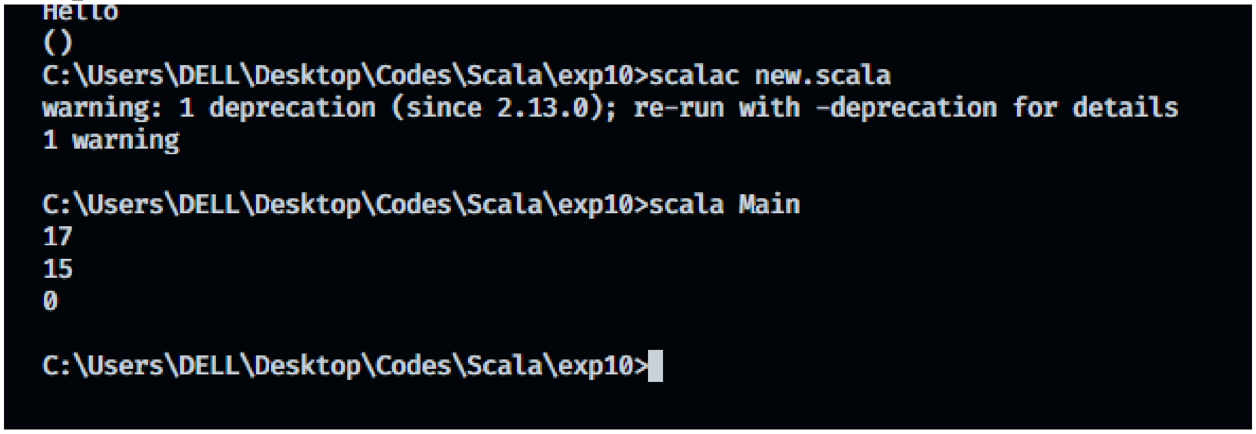


```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.sca
warning: 1 deprecation (since 2.13.0); re-run with -d
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Hello
()
C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

2.
Code :

```
object Main {
   def main(args: Array[String]) = {
        var result1 = functionExample(15,2)
        var result2 = functionExample(15)
        var result3 = functionExample()
        println(result1+"\n"+result2+"\n"+result3)
    }
    def functionExample(a:Int = 0, b:Int = 0):Int = {
        a+b
    }
}
```

Output:

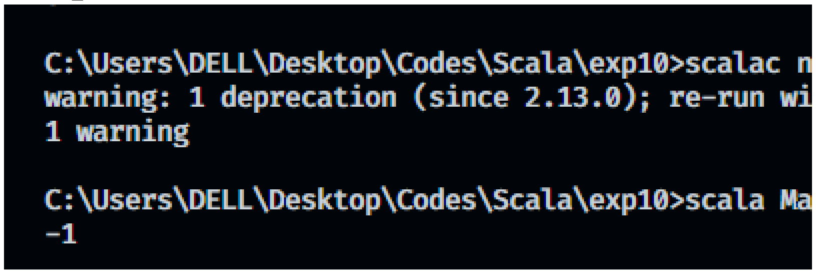


```
Hello
()
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
17
15
0

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

3.
Code :

```
object Main {
   def main(args: Array[String]) {
      val result = checkIt(-10)
      println (result)
   }
    def checkIt (a:Int)  =  if (a >= 0) 1 else -1
}
```

Output:

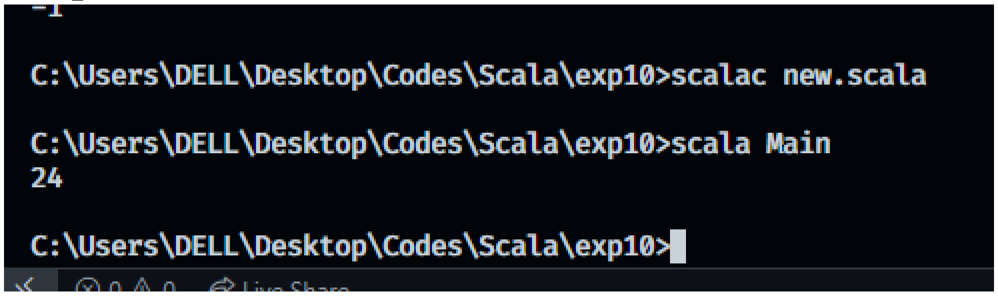


```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac n
warning: 1 deprecation (since 2.13.0); re-run wi
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Ma
-1
```

4.
Code :

```
object Main {
   def main(args: Array[String]) = {
     var result = multiplyBy2(add2(10))
     println(result)
    }
    def add2(a:Int):Int = {
        a+2
    }

    def multiplyBy2(a:Int):Int = {
        a*2
    }
}
```

Output:

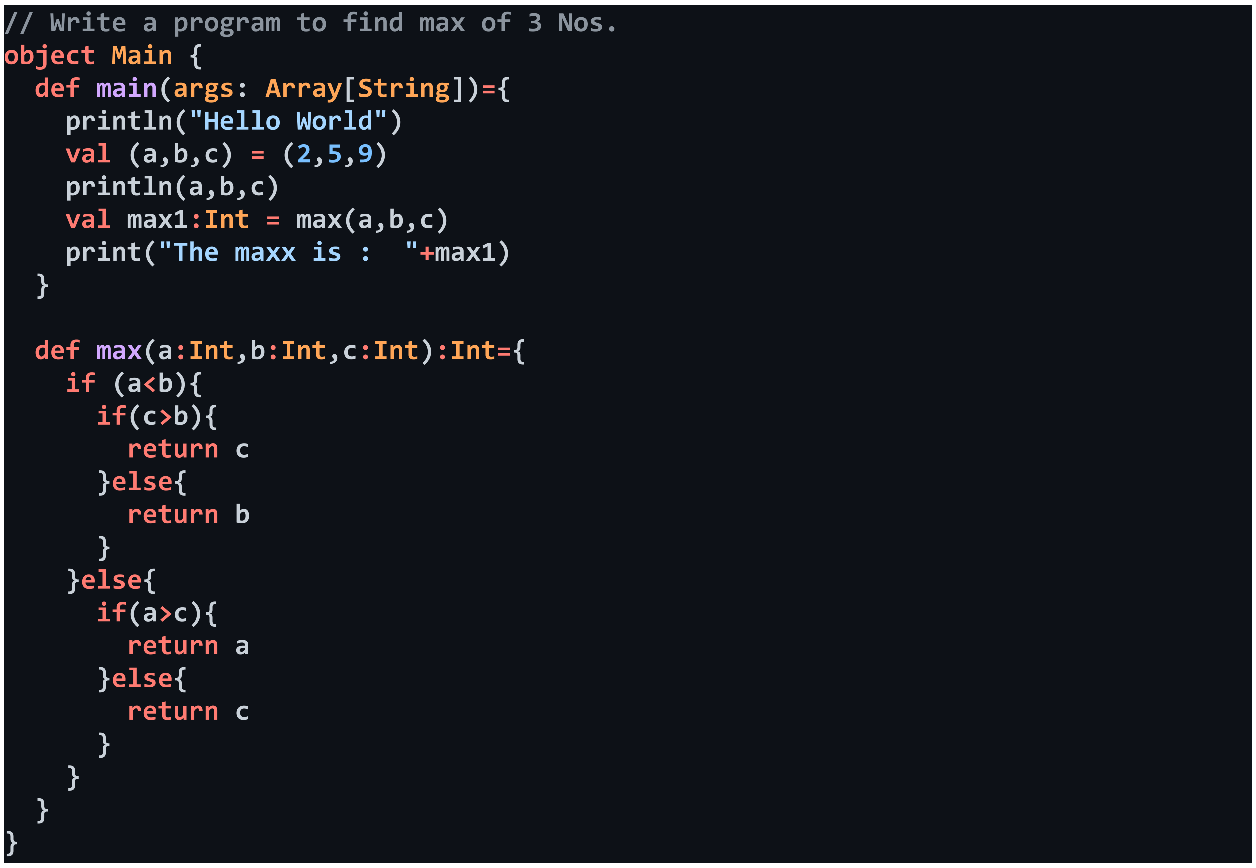


```
-1

C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
24

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

SAPID : 60009200030 – Sarvagya Singh – K1
Scala Experiment – lab 8

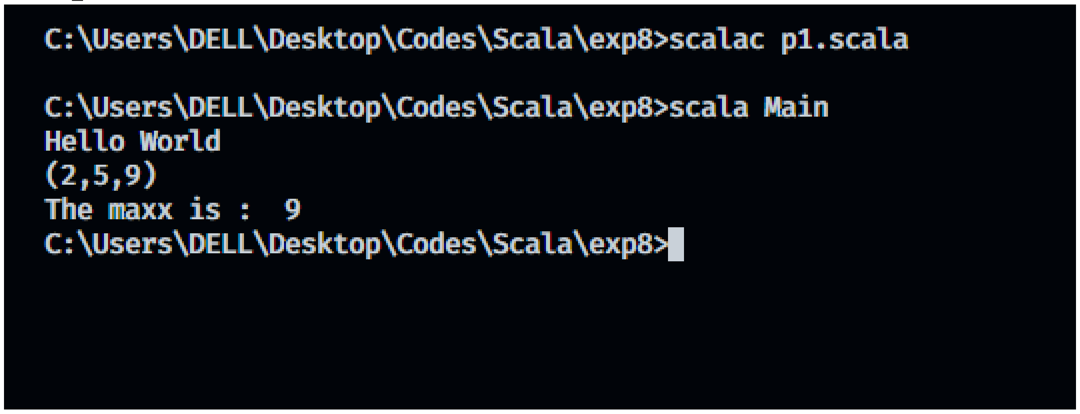
1.Write a program to find max of 3 Nos.

Code :



```
// Write a program to find max of 3 Nos.
object Main {
  def main(args: Array[String])={
    println("Hello World")
    val (a,b,c) = (2,5,9)
    println(a,b,c)
    val max1:Int = max(a,b,c)
    print("The maxx is :  "+max1)
  }

  def max(a:Int,b:Int,c:Int):Int={
    if (a<b){
      if(c>b){
        return c
      }else{
        return b
      }
    }else{
      if(a>c){
        return a
      }else{
        return c
      }
    }
  }
}
```

Output:



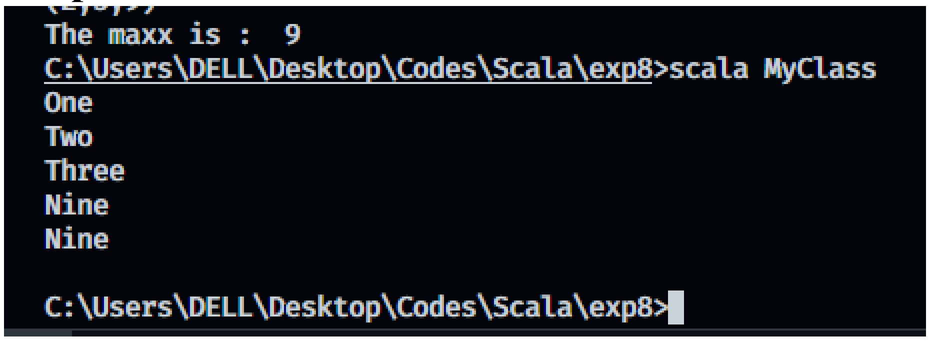
```
C:\Users\DELL\Desktop\Codes\Scala\exp8>scalac p1.scala

C:\Users\DELL\Desktop\Codes\Scala\exp8>scala Main
Hello World
(2,5,9)
The maxx is :  9
C:\Users\DELL\Desktop\Codes\Scala\exp8>
```

2.Write a program to print given no in words using pattern matching and while loop .eg 123 output one two three.

Code :

```
object MyClass {
def max(x:Int, y:Int, z:Int)=if(x > y && x > z) x else if(y > x && y > z) y else
z
def patternMatch(x:String){
for(i<- x){
i match{
case '1' => println("One")
case '2' => println("Two")
case '3' => println("Three")
case '4' => println("Four")
case '5' => println("Five")
case '6' => println("Six")
case '7' => println("Seven")
case '8' => println("Eight")
case '9' => println("Nine")
case '0' => println("Zero")
case _ => println("Default")
}
}
}
def main(args: Array[String]) {
patternMatch("12399");
}
}
```

Output:

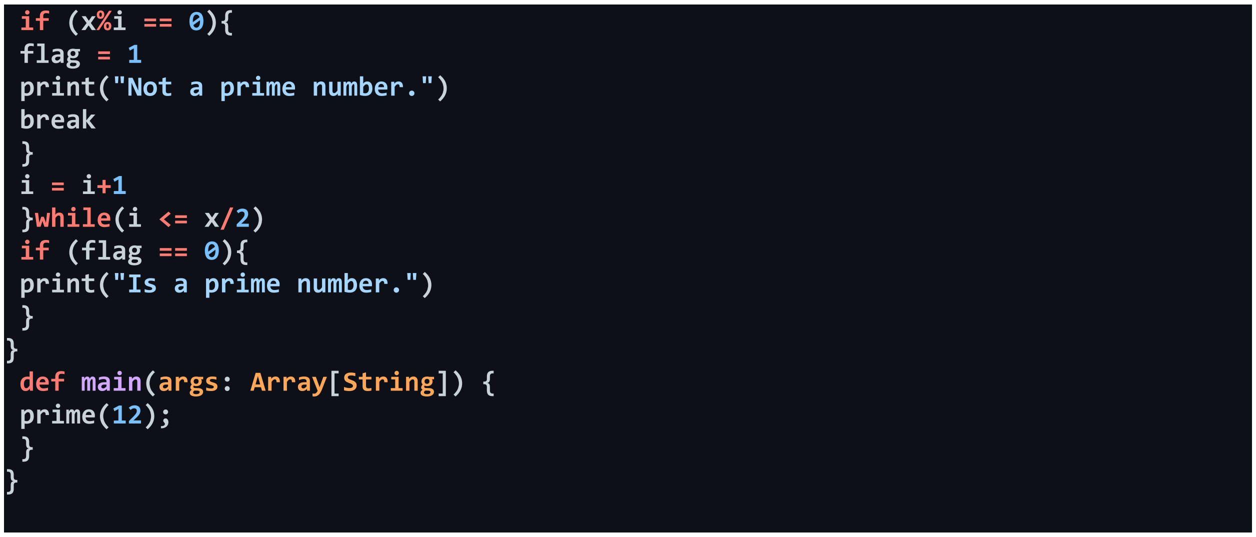


```
The maxx is :  9
C:\Users\DELL\Desktop\Codes\Scala\exp8>scala MyClass
One
Two
Three
Nine
Nine

C:\Users\DELL\Desktop\Codes\Scala\exp8>
```

3.Write a program to find whether the no is prime or not using do while loop.

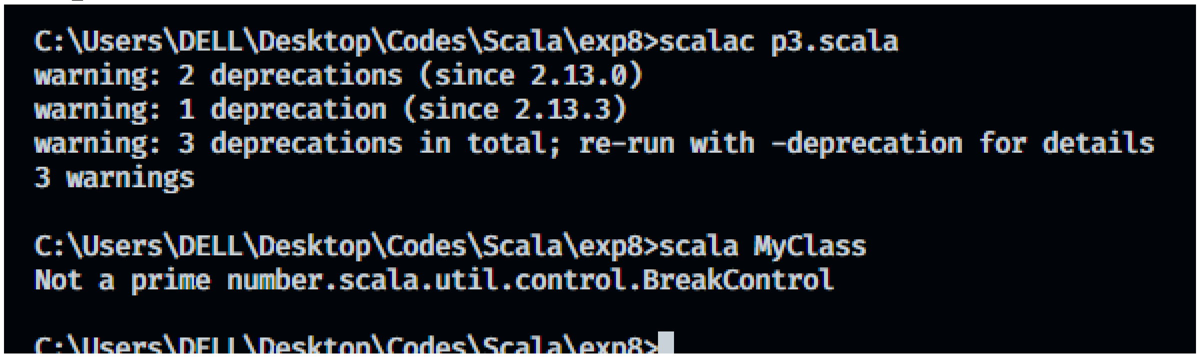
Code :

```
import scala.util.control.Breaks._
object MyClass {
 def prime(x:Int){
 var flag = 0
 var i = 2
 do{
```



```
 if (x%i == 0){
 flag = 1
 print("Not a prime number.")
 break
 }
 i = i+1
 }while(i <= x/2)
 if (flag == 0){
 print("Is a prime number.")
 }
}
 def main(args: Array[String]) {
 prime(12);
 }
}
```

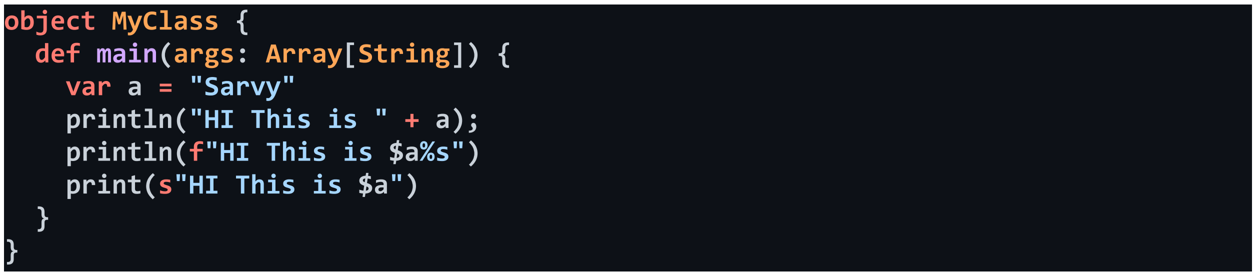
Output :



```
C:\Users\DELL\Desktop\Codes\Scala\exp8>scalac p3.scala
warning: 2 deprecations (since 2.13.0)
warning: 1 deprecation (since 2.13.3)
warning: 3 deprecations in total; re-run with -deprecation for details
3 warnings

C:\Users\DELL\Desktop\Codes\Scala\exp8>scala MyClass
Not a prime number.scala.util.control.BreakControl

C:\Users\DELL\Desktop\Codes\Scala\exp8>
```

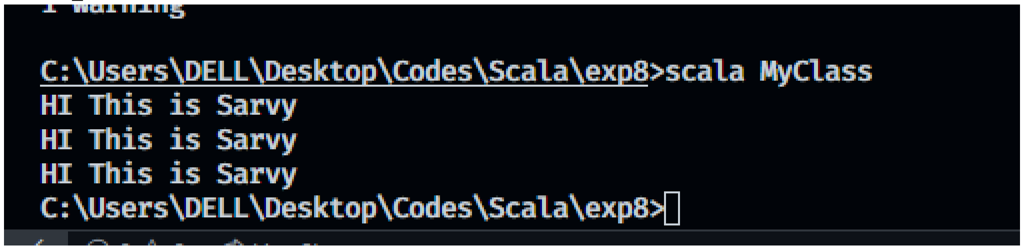
4.Write a program in Scala to demonstrate string interpolation.

Code :



```
object MyClass {
  def main(args: Array[String]) {
    var a = "Sarvy"
    println("HI This is " + a);
    println(f"HI This is $a%s")
    print(s"HI This is $a")
  }
}
```

Output :



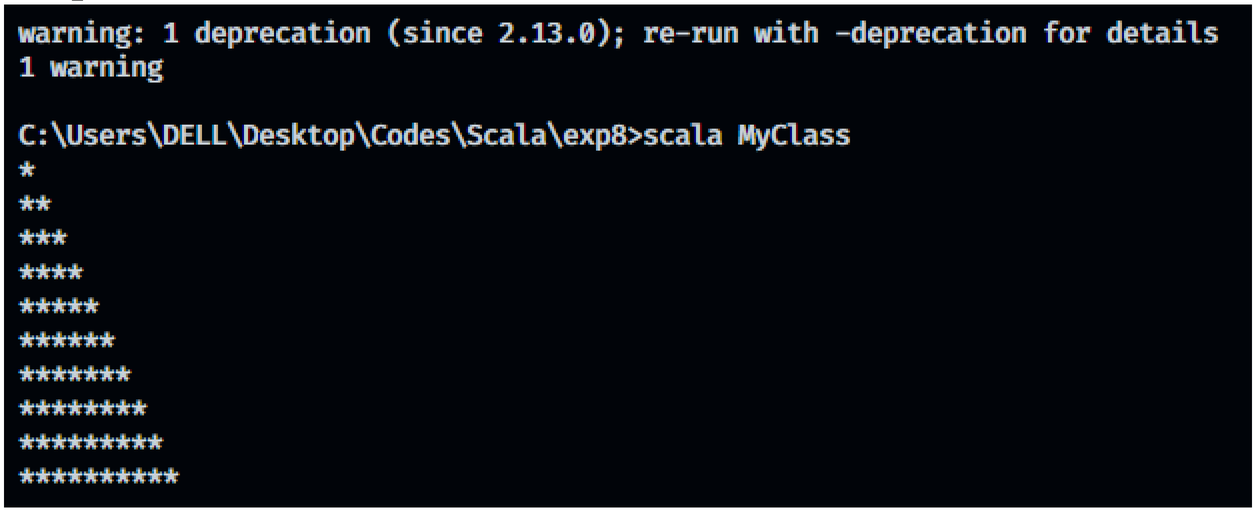
```
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp8>scala MyClass
HI This is Sarvy
HI This is Sarvy
HI This is Sarvy
C:\Users\DELL\Desktop\Codes\Scala\exp8>
```

5. Write a program that prints the following patterns.

Code :

```
object MyClass {
 def main(args: Array[String]) {
  var i = 1;
  var j=0;
  for(i<-1 to 10){
  for(j<-1 to i){
    print("*");
  }
  println();
 }
 }
}
```

Output :



```
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp8>scala MyClass
*
**
***
****
*****
******
*******
********
*********
**********
```

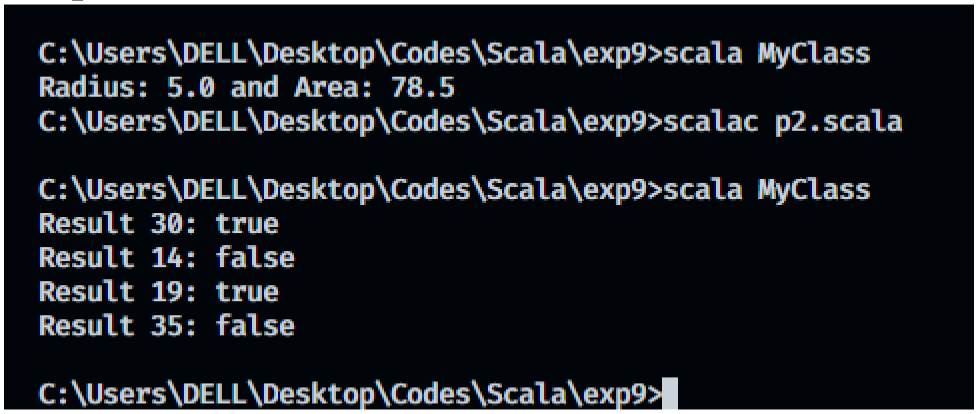
Scala Experiment – lab 9

1.Write a Scala program to check whether a given positive number is a multiple of 3 or a multiple of 7

Code :

```
object MyClass {
  def test(n: Int): Boolean =
    {
    n % 3 == 0 || n % 7 == 0;
    }

   def main(args: Array[String]): Unit = {
      println("Result 30: " + test(30));
      println("Result 14: " + test(19));
      println("Result 19: " + test(21));
      println("Result 35: " + test(10));
    }
}
```

Output:

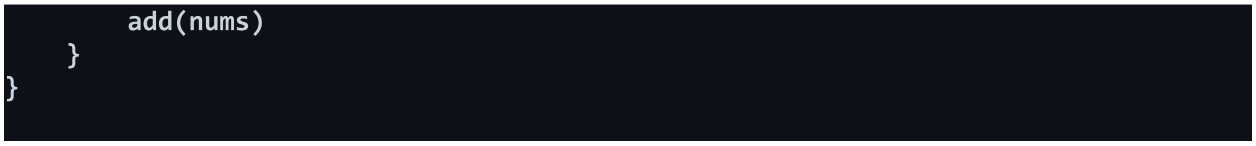


```
C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
Radius: 5.0 and Area: 78.5
C:\Users\DELL\Desktop\Codes\Scala\exp9>scalac p2.scala

C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
Result 30: true
Result 14: false
Result 19: true
Result 35: false

C:\Users\DELL\Desktop\Codes\Scala\exp9>
```

2.Write a Scala program to find sum of square of the given list.

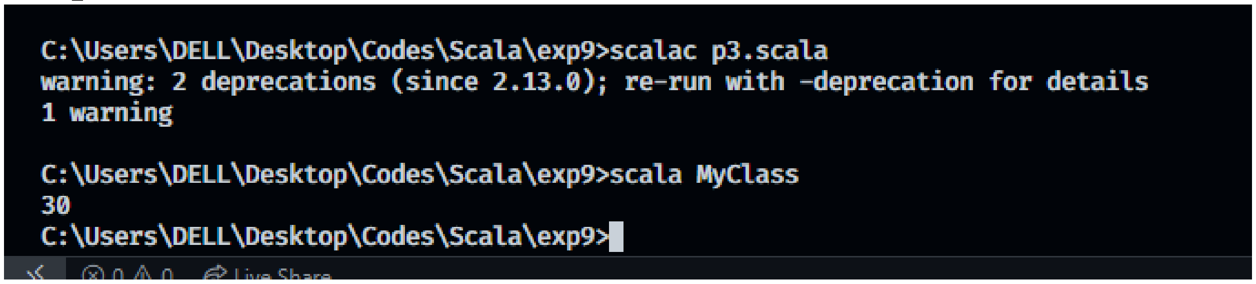
Code :

```
object MyClass {
    var sum = 0
    def add(a:List[Int]){
        for (i <- a){
            sum += i*i
        }
        print(sum)
    }

    def main(args: Array[String]) {
        var nums: List[Int] = List(1, 2, 3, 4)
```



```
        add(nums)
    }
}
```

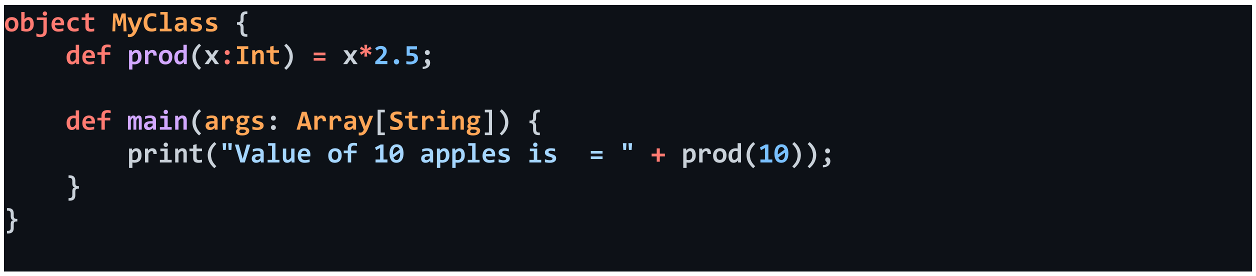
Output:



```
C:\Users\DELL\Desktop\Codes\Scala\exp9>scalac p3.scala
warning: 2 deprecations (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
30
C:\Users\DELL\Desktop\Codes\Scala\exp9>
```

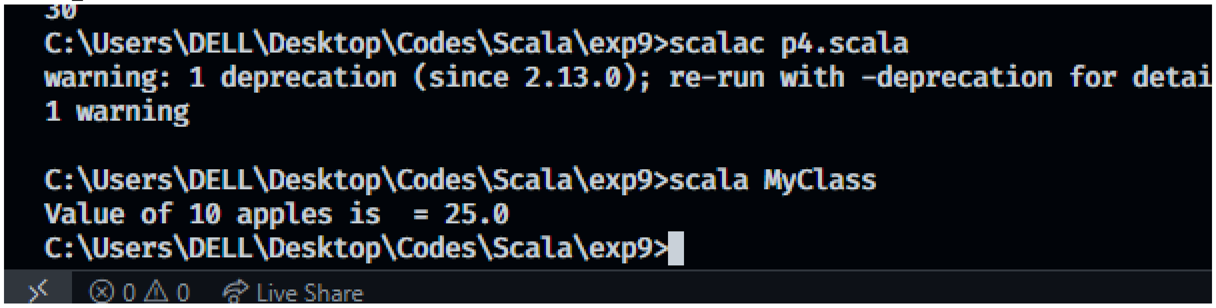
3.Write a Scala program to calculate the total cost for a customer who is buying 10 Glazed donuts. You can assume that the price of each Glazed donut item is at $2.50.

Code :



```
object MyClass {
    def prod(x:Int) = x*2.5;

    def main(args: Array[String]) {
        print("Value of 10 apples is  = " + prod(10));
    }
}
```

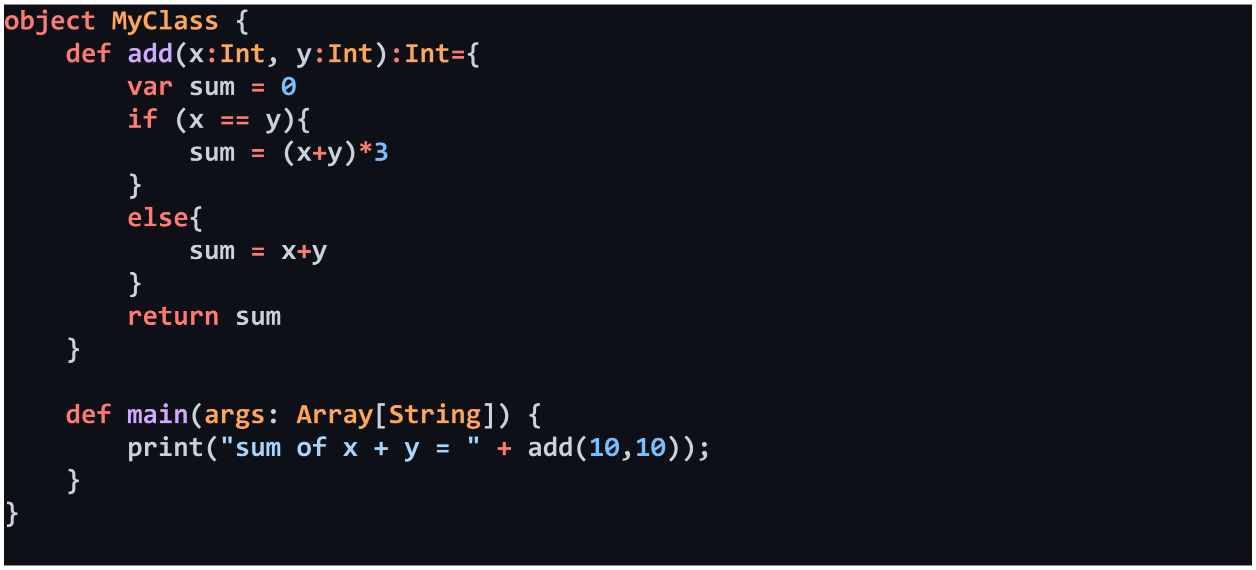
Output:



```
30
C:\Users\DELL\Desktop\Codes\Scala\exp9>scalac p4.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for detai
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
Value of 10 apples is  = 25.0
C:\Users\DELL\Desktop\Codes\Scala\exp9>
```

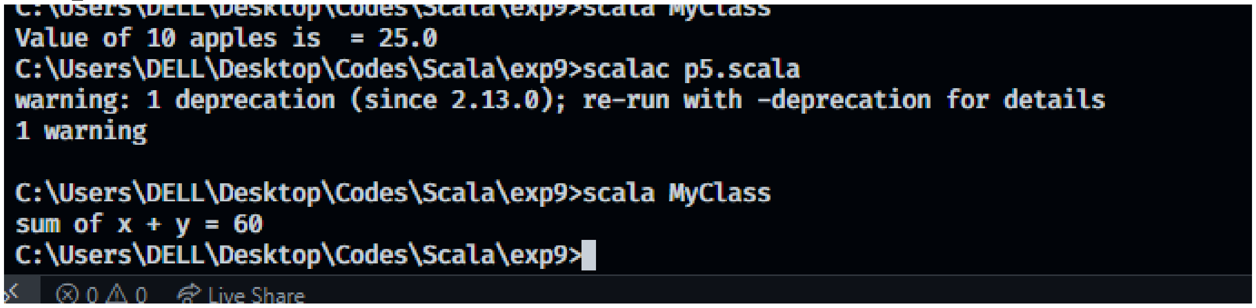
4.Write a Scala program to compute the sum of the two given integer values. If the two values are the same, then return triples their sum.

Code :



```scala
object MyClass {
    def add(x:Int, y:Int):Int={
        var sum = 0
        if (x == y){
            sum = (x+y)*3
        }
        else{
            sum = x+y
        }
        return sum
    }

    def main(args: Array[String]) {
        print("sum of x + y = " + add(10,10));
    }
}
```

Output :

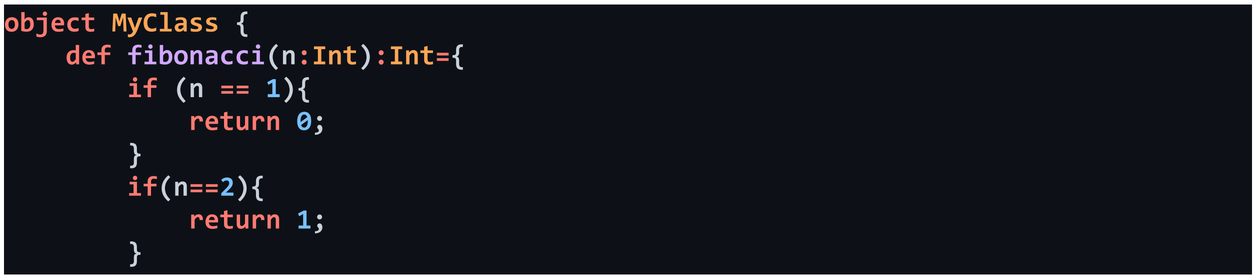


```
C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
Value of 10 apples is  = 25.0
C:\Users\DELL\Desktop\Codes\Scala\exp9>scalac p5.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
sum of x + y = 60
C:\Users\DELL\Desktop\Codes\Scala\exp9>
```

5) Write a recursive function to get the nth Fibonacci number. The first two Fibonacci numbers are 0 and 1. The nth number is always the sum of the previous two—the sequence begins 0, 1, 1, 2, 3, 5.

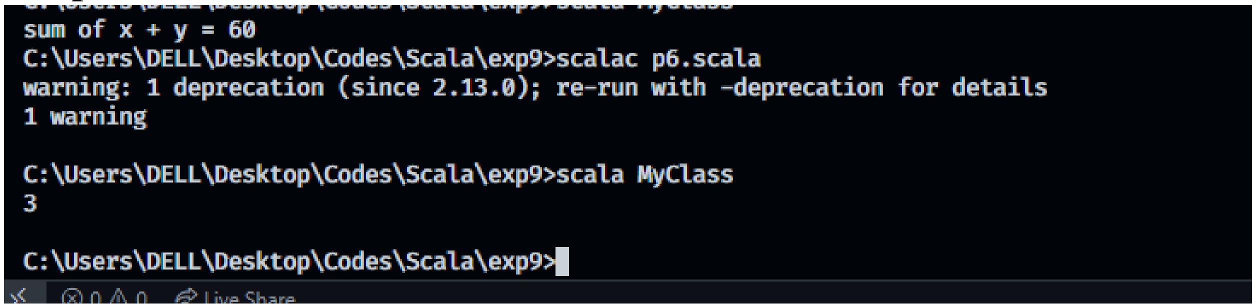
def fib (n: Int): Int

Code :



```scala
object MyClass {
    def fibonacci(n:Int):Int={
        if (n == 1){
            return 0;
        }
        if(n==2){
            return 1;
        }
        return fibonacci(n - 1) + fibonacci(n - 2);
    }

    def main(args: Array[String]) {
        println(fibonacci(5))
    }
}
```

Output:



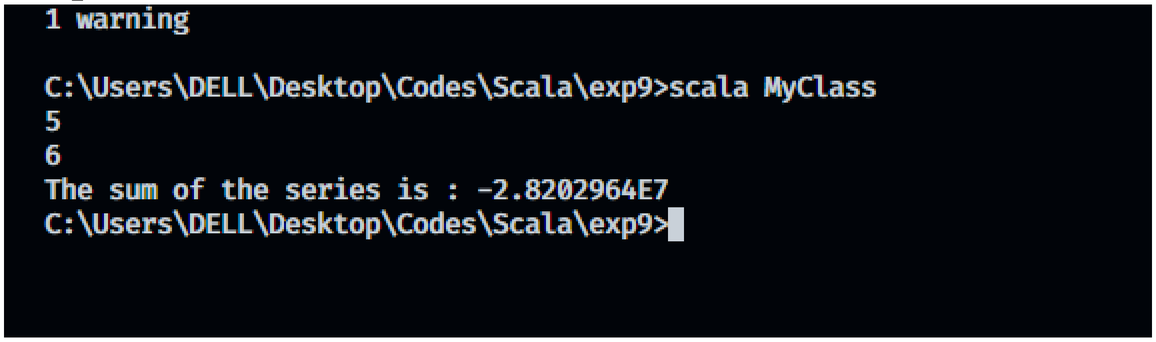
```
sum of x + y = 60
C:\Users\DELL\Desktop\Codes\Scala\exp9>scalac p6.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
3

C:\Users\DELL\Desktop\Codes\Scala\exp9>
```

6) Write a function to find the values of following series: -Value=a+a2/2! +a3/3! +a4+4!................an/n!

Code:

```
object MyClass {
    def fact(x:Int):Int={
        if(x==0 || x==1){
            return 1;
        }
        else{
            return x * fact(x-1);
        }
    }

    def power(x:Int,y:Int):Int={
        var pro = x;
        var i = y;
        while(i!=0){
            pro *= pro;
            i-=1;
        }
        return pro;
    }


    def series_sum(a:Int,x:Int,fact:Int=>Int,power:(Int,Int)=>Int):Double={
        var exp = 0.0;
        for(i <- 1 to a){
            exp = exp + power(x,i)/fact(i);
        }
        return exp
    }

    def main(args: Array[String]) {
        val a = scala.io.StdIn.readInt();
        val x = scala.io.StdIn.readInt();
        print("The sum of the series is : ");
        print(series_sum(a,x,fact,power))
    }
}
```

Output:



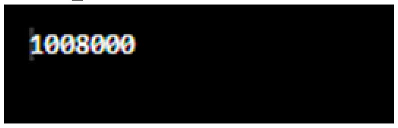
```
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp9>scala MyClass
5
6
The sum of the series is : -2.8202964E7
C:\Users\DELL\Desktop\Codes\Scala\exp9>
```

7.Write a function to find multiplication of first 10 no's using function with variable length parameters

Code :

```
def mult(args: Int*) = {
    var mul = 1;
    for(a <- args) mul*=a
    mul
}
var mul = mult(2,2,7,2,3,4,5,6,10,5);
println(mul);
```

Output:



```
1008000
```

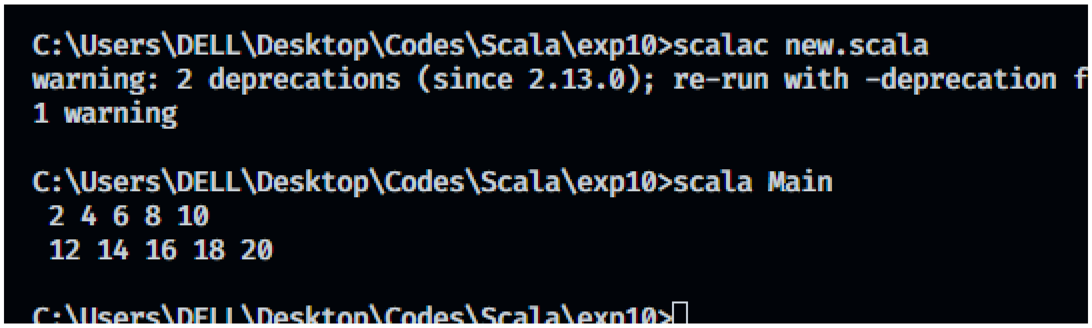
SAPID : 60009200030 – Sarvagya Singh – K1
Scala Experiment – lab 10

1.Write a program to make a class called as Circle. It should have three methods namely: accept radius, calculate area and display the area.

Code :

```scala
class ArrayExample{

    var arr1 = Array(Array(1,2,3,4,5), Array(6,7,8,9,10))

    var arr2 = Array(Array(1,2,3,4,5), Array(6,7,8,9,10))

    var arr3 = Array.ofDim[Int](2,5)

    def show(){

        for(i<- 0 to 1){

           for(j<- 0 to 4){

                arr3(i)(j) = arr1(i)(j)+arr2(i)(j)

                print(" "+arr3(i)(j))

            }

            println()

        }

    }

}

object Main{

    def main(args:Array[String]){

        var a = new ArrayExample()

        a.show()

    }
}
```

Output:

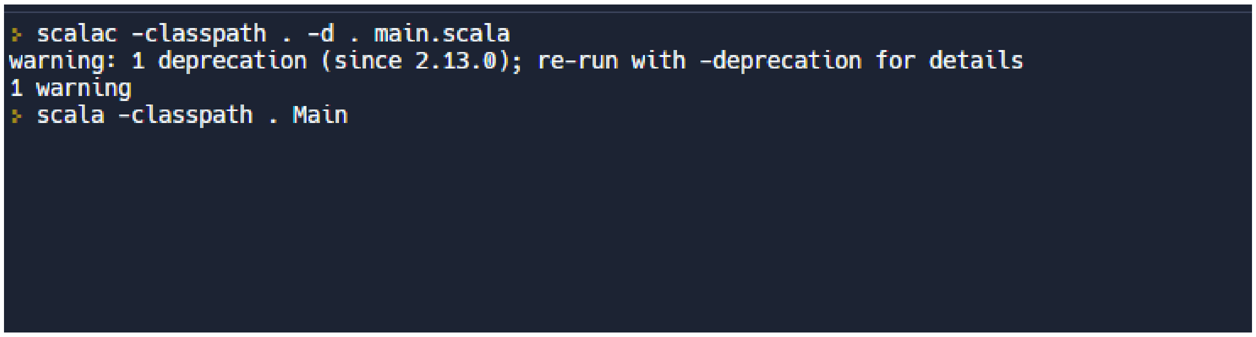


```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 2 deprecations (since 2.13.0); re-run with -deprecation f
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
 2 4 6 8 10
 12 14 16 18 20

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

2.Code :

```
object Main {

    def sumS():Int={

        var sum = 0;

        for(i <- 1 to 10){

            sum = sum + i*i

        }

        return sum;

    }

    def main(args: Array[String]) {

        println("the sum of sqaures from 1 to 10 is " + sumS());

    }

}
```

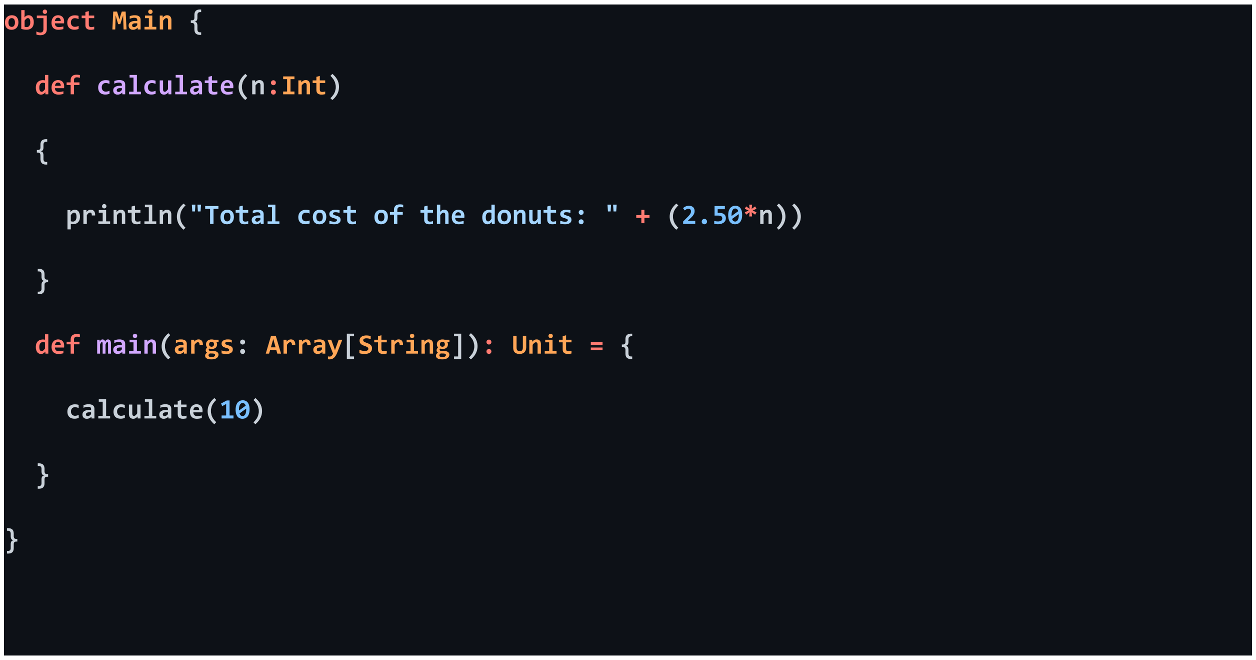
Output:



```
> scalac -classpath . -d . main.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning
> scala -classpath . Main
```

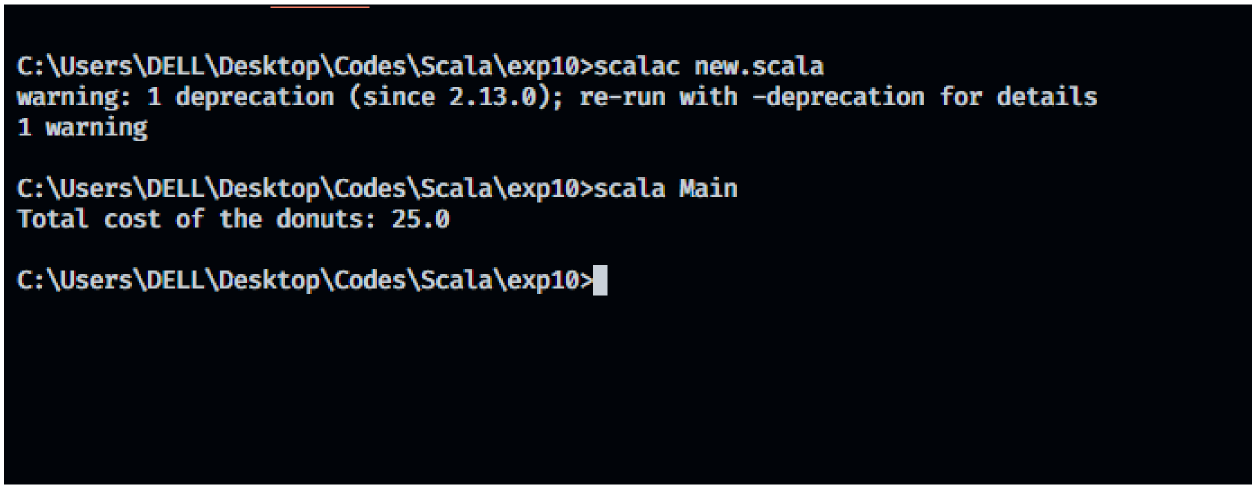
3.Show employee () to display employee details.

Code :



```
object Main {

  def calculate(n:Int)

  {

    println("Total cost of the donuts: " + (2.50*n))

  }

  def main(args: Array[String]): Unit = {

    calculate(10)

  }

}
```

Output:



```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Total cost of the donuts: 25.0

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

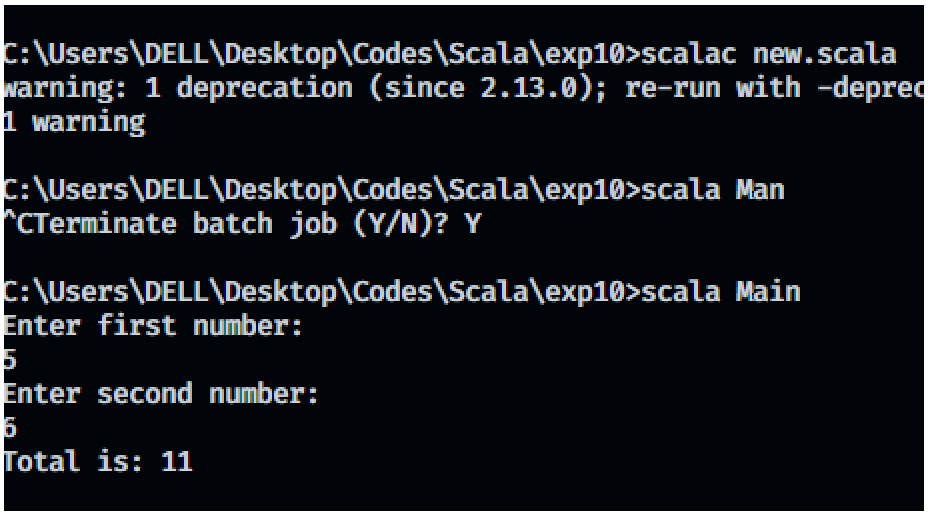
4.

Code :

```
import java.util.Scanner

object Main {

  def calculate_sum(n1:Int, n2:Int)

  {

    if(n1==n2)

    {

      println("Total is: " + (n1+n2)*3)

    }

    else

    {

      println("Total is: " + (n1+n2))

    }

  }

  def main(args: Array[String]): Unit = {

    var s = new Scanner(System.in)
```

```scala
    var n1:Int = 0

    var n2:Int = 0

    println("Enter first number: ")

    n1 = s.nextInt()

    println("Enter second number: ")

    n2 = s.nextInt()

    calculate_sum(n1, n2)

  }

}
```

Output:



```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprec
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Man
^CTerminate batch job (Y/N)? Y

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Enter first number:
5
Enter second number:
6
Total is: 11
```

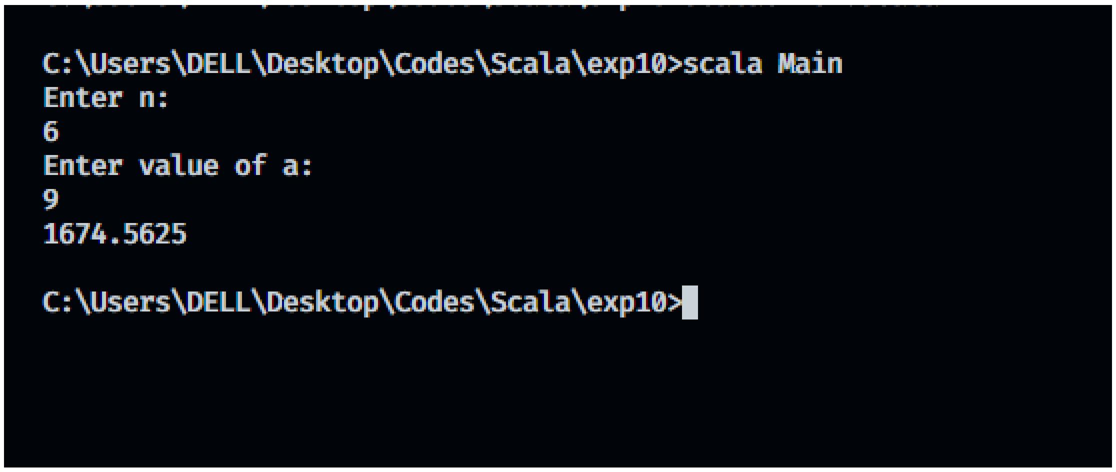
5.

Code :

```scala
import java.util.Scanner
```

```
object  Main
{
  def factorial(n:Int):Int =
  {
    var f = 1
    for(i <- 1 to n)
    {
      f = f*i
    }
    return f
  }

  def series(a:Int, n:Int):Double =
  {
    var sum:Double = 0
    for(i <- 1 to n)
    {
      sum += (Math.pow(a, i)/factorial(i))
    }
    return sum
  }
  def main(args:Array[String]):Unit={
    var s = new Scanner(System.in)
    var n:Int = 0
    var a:Int = 0
    var total:Double = 0
    println("Enter n: ")
    n = s.nextInt()

    println("Enter value of a: ")

    a = s.nextInt()

    total = series(a, n)

    println(total)

  }

}
```

Output:



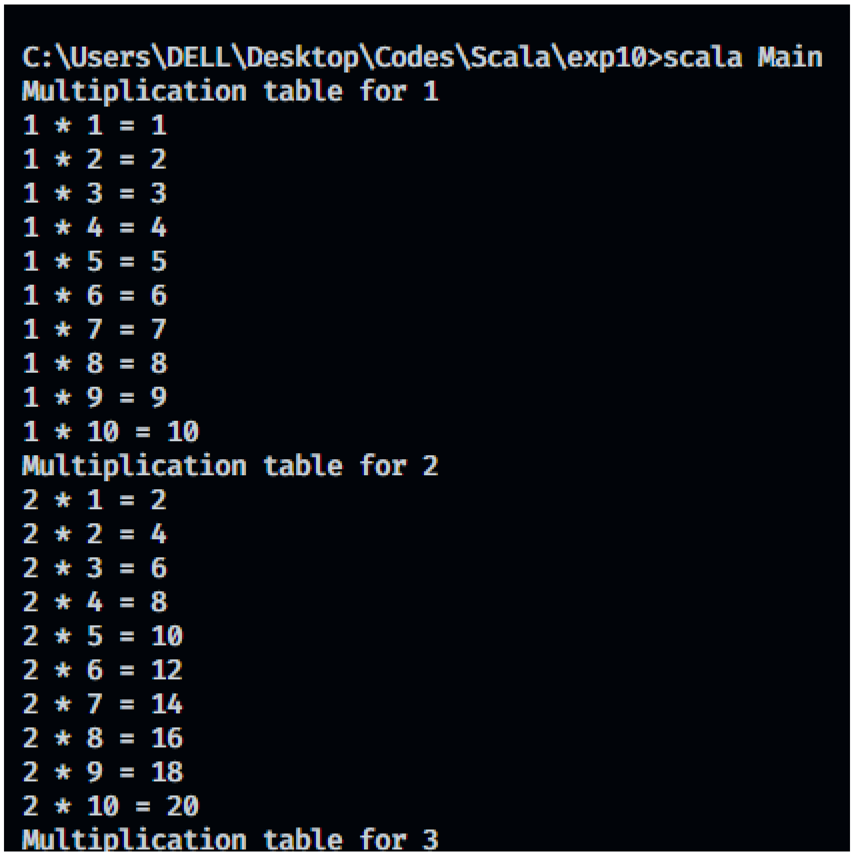
```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Enter n:
6
Enter value of a:
9
1674.5625

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

6.

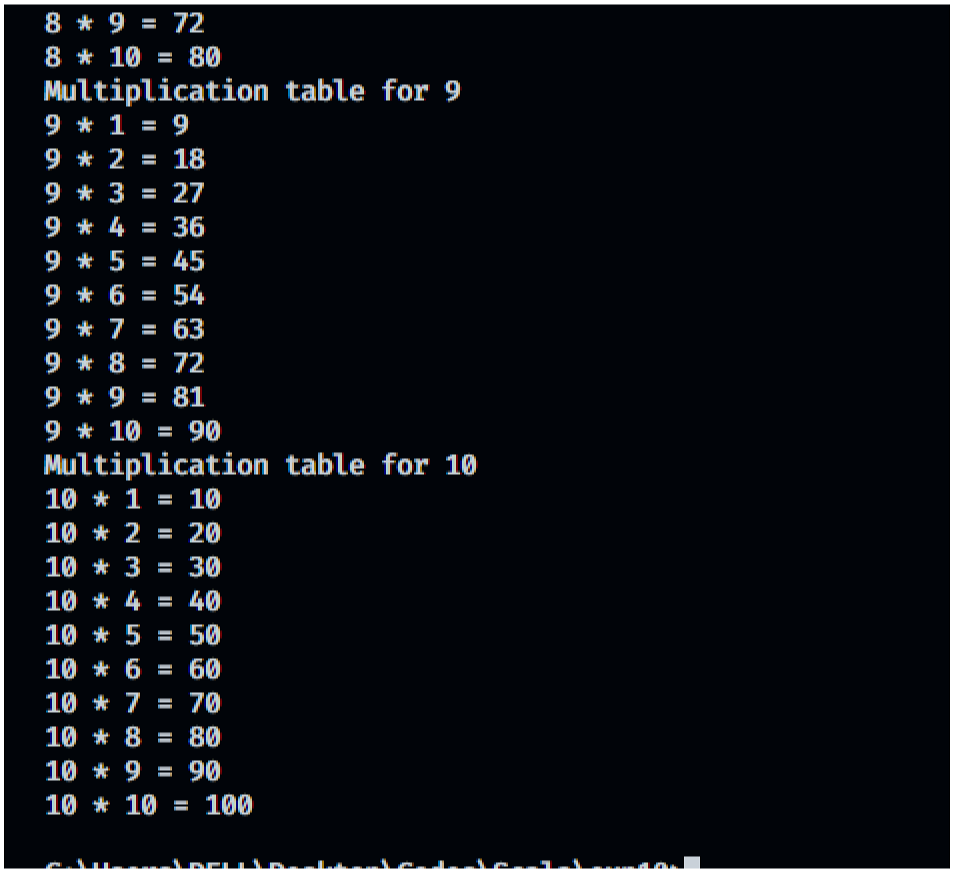
Code :

```
object Main

{

  def multiplication(n:Int)

  {

    for(i <- 1 to n)

    {

      println("Multiplication table for " + i)
      for(j <- 1 to 10)
      {
        println(i + " * " + j + " = " + (i*j))
      }
    }
  }
  def main(args:Array[String]):Unit={
    multiplication(10)
  }
}
```

Output:



```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Multiplication table for 1
1 * 1 = 1
1 * 2 = 2
1 * 3 = 3
1 * 4 = 4
1 * 5 = 5
1 * 6 = 6
1 * 7 = 7
1 * 8 = 8
1 * 9 = 9
1 * 10 = 10
Multiplication table for 2
2 * 1 = 2
2 * 2 = 4
2 * 3 = 6
2 * 4 = 8
2 * 5 = 10
2 * 6 = 12
2 * 7 = 14
2 * 8 = 16
2 * 9 = 18
2 * 10 = 20
Multiplication table for 3
```



```
8 * 9 = 72
8 * 10 = 80
Multiplication table for 9
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
9 * 10 = 90
Multiplication table for 10
10 * 1 = 10
10 * 2 = 20
10 * 3 = 30
10 * 4 = 40
10 * 5 = 50
10 * 6 = 60
10 * 7 = 70
10 * 8 = 80
10 * 9 = 90
10 * 10 = 100

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

SAPID : 60009200030 – Sarvagya Singh – K1

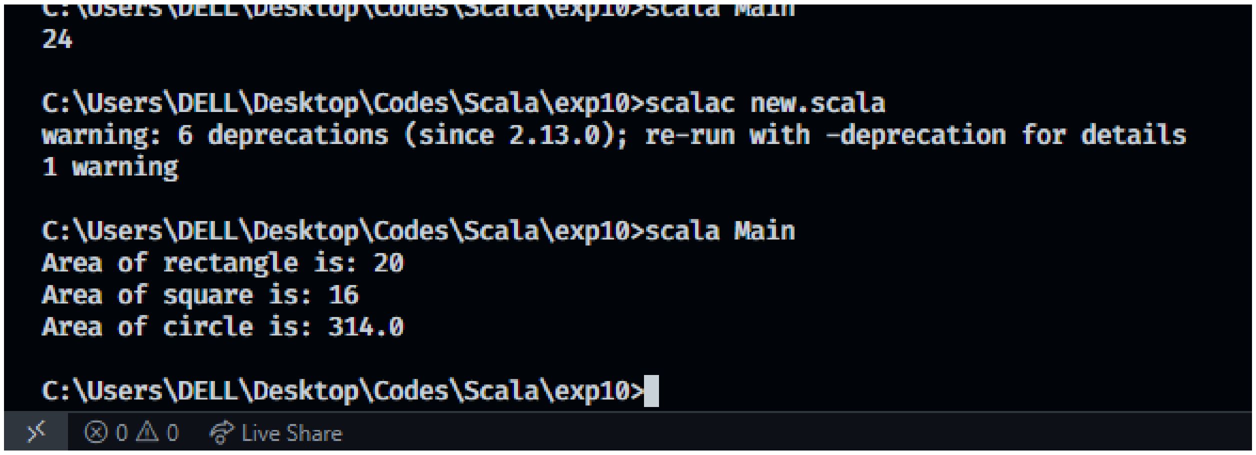
Scala Experiment – lab 11

1.

Code :

```scala
abstract class Shape
{
  def RectangleArea(length:Int, breadth:Int)
  {
    println("Area of rectangle is: " + (length*breadth))
  }

  def SquareArea(side:Int)
  {
    println("Area of square is: " + (side*side))
  }

  def CircleArea(radius:Int)
  {
    println("Area of circle is: " + (3.14*radius*radius))
  }
}

class Area
{
  def RectangleArea(length:Int, breadth:Int)

  {

    println("Area of rectangle is: " + (length*breadth))

  }


  def SquareArea(side:Int)

  {

    println("Area of square is: " + (side*side))

  }


  def CircleArea(radius:Int)

  {

    println("Area of circle is: " + (3.14*radius*radius))

  }

}


object Main {

  def main(args: Array[String]): Unit = {

    var a = new Area()

    a.RectangleArea(4, 5)

    a.SquareArea(4)

    a.CircleArea(10)

  }

}
```

Output:



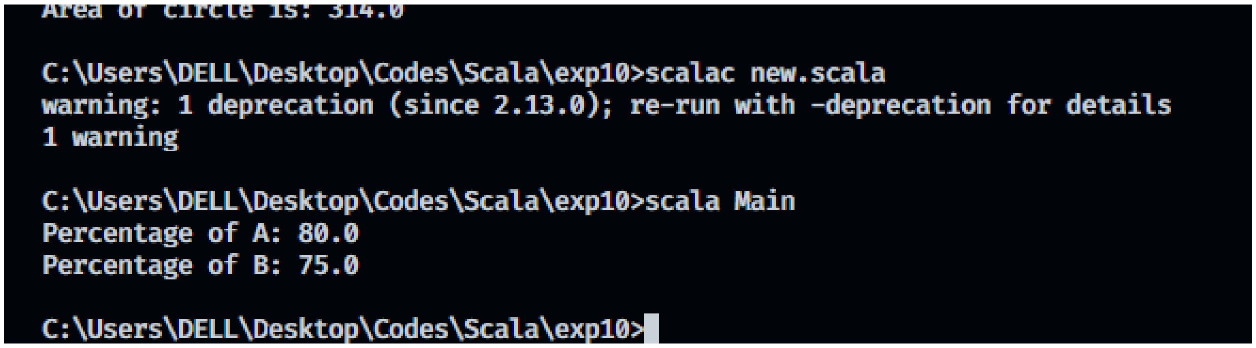
```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
24

C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 6 deprecations (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Area of rectangle is: 20
Area of square is: 16
Area of circle is: 314.0

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

2.

Code :

```
abstract class Marks

{

  def getPercentage():Double

}


class A(p:Int, c:Int, m:Int) extends Marks

{

  def getPercentage() :Double=

  {

    var sum = this.p + this.c + this.m

    return (sum/3)

  }

}


class B(p:Int, c:Int, m:Int, b:Int) extends Marks

{
  def getPercentage():Double=

  {

    var sum = this.p + this.c + this.b + this.m

    return (sum/4)

  }

}


object Main

{

  def main(args:Array[String]){

    var a = new A(70, 80, 90)

    var b = new B(60, 70, 80, 90)

    println("Percentage of A: " + a.getPercentage())

    println("Percentage of B: " + b.getPercentage())

  }

}
```

Output:



```
Area of circle is: 314.0

C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Percentage of A: 80.0
Percentage of B: 75.0

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

3.

Code :

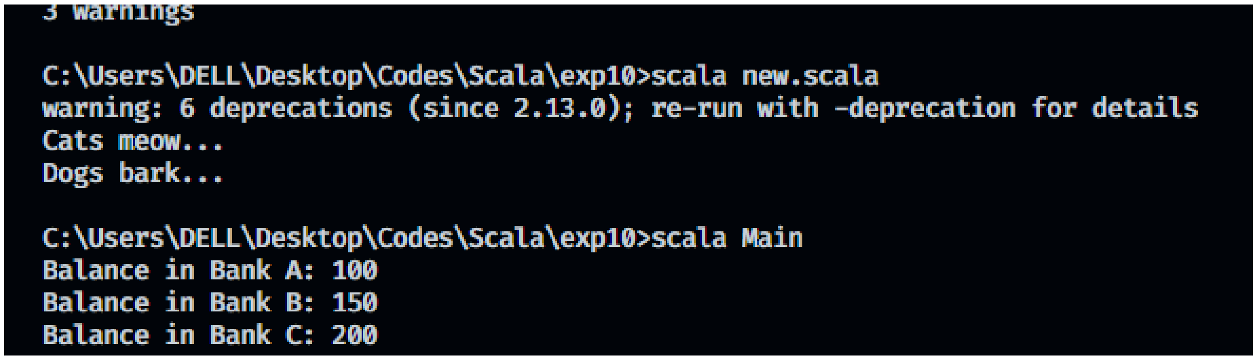
```
abstract class Bank

{

  def getBalance:Int

}


class BankA(a:Int) extends Bank

{

  def getBalance():Int={

    return a

  }

}


class BankB(a:Int) extends Bank

{

  def getBalance():Int={

    return a

  }

}


class BankC(a:Int) extends Bank

{

  def getBalance():Int={

    return a

  }
}


object Main
{
  def main(args:Array[String]):Unit={
    var a = new BankA(100)
    var b = new BankB(150)
    var c = new BankC(200)
    println("Balance in Bank A: " + a.getBalance())
    println("Balance in Bank B: " + b.getBalance())
    println("Balance in Bank C: " + c.getBalance())
  }
}
```

4.

Code :

```
abstract class Animals
{
  def cats()
  def dogs()
}


class Cats extends Animals
{
  def cats()
  {
    println("Cats meow...")
  }
  def dogs()
  {

  }
}

class Dogs extends Animals
{
  def cats()
  {

  }
  def dogs()
  {
    println("Dogs bark...")
  }
}

object Main
{
  def main(args:Array[String]):Unit={
    var c = new Cats()
    var d = new Dogs()

    c.cats()

    d.dogs()

  }

}
```

Output:



```
3 warnings

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala new.scala
warning: 6 deprecations (since 2.13.0); re-run with -deprecation for details
Cats meow...
Dogs bark...

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Balance in Bank A: 100
Balance in Bank B: 150
Balance in Bank C: 200
```

SAPID : 60009200030 – Sarvagya Singh – K1

Scala Experiment – lab 12

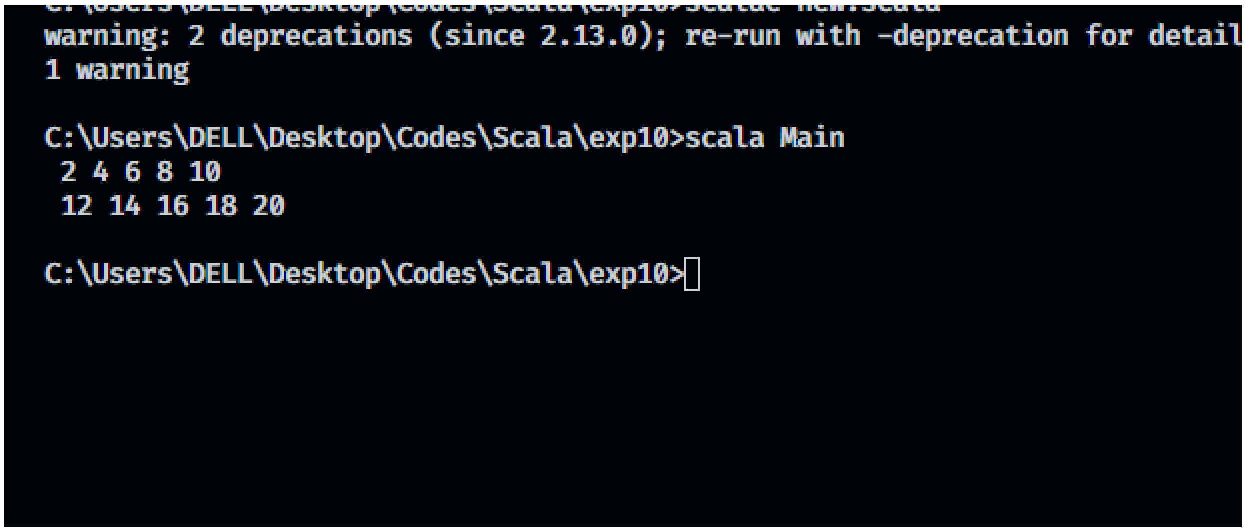
1.

Code :

```
class ArrayExample{
    var arr1 = Array(Array(1,2,3,4,5), Array(6,7,8,9,10))
    var arr2 = Array(Array(1,2,3,4,5), Array(6,7,8,9,10))
    var arr3 = Array.ofDim[Int](2,5)
    def show(){
        for(i<- 0 to 1){
           for(j<- 0 to 4){
                arr3(i)(j) = arr1(i)(j)+arr2(i)(j)
                print(" "+arr3(i)(j))
            }
            println()
        }
    }
}
```

```scala
object Main{

    def main(args:Array[String]){

        var a = new ArrayExample()

        a.show()

    }

}
```

Output:



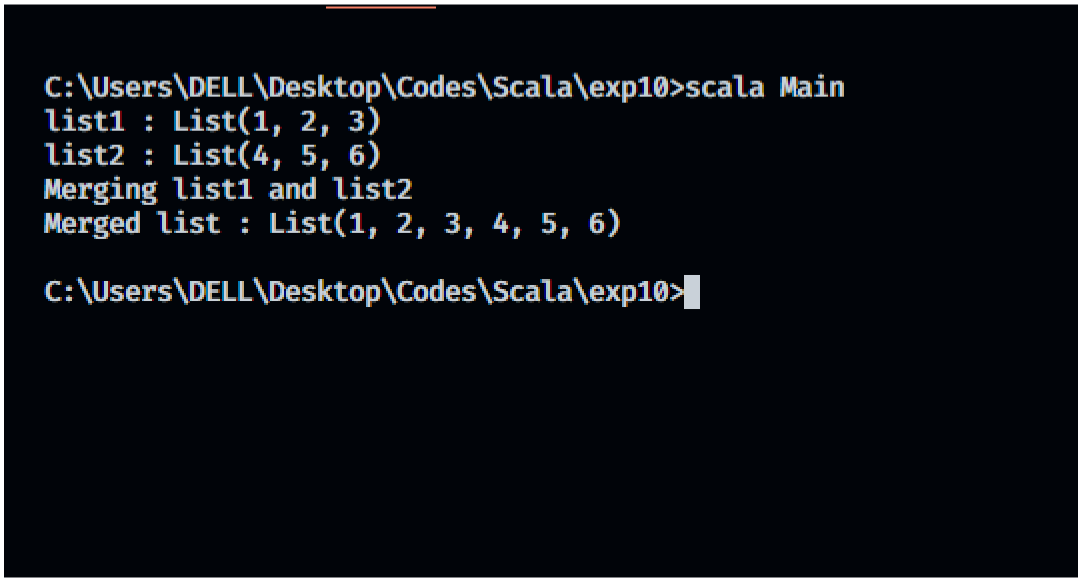
```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 2 deprecations (since 2.13.0); re-run with -deprecation for detail
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
 2 4 6 8 10
 12 14 16 18 20

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

2.

Code :

```scala
object Main {

    def main(args: Array[String]) {

        val list1 = List("1", "2", "3")

        val list2 = List("4", "5", "6")


        println("list1 : " + list1)
        println("list2 : " + list2)


        println("Merging list1 and list2 ")


        val list3 = list1 ++ list2


        println("Merged list : " + list3)

    }

}
```

Output:

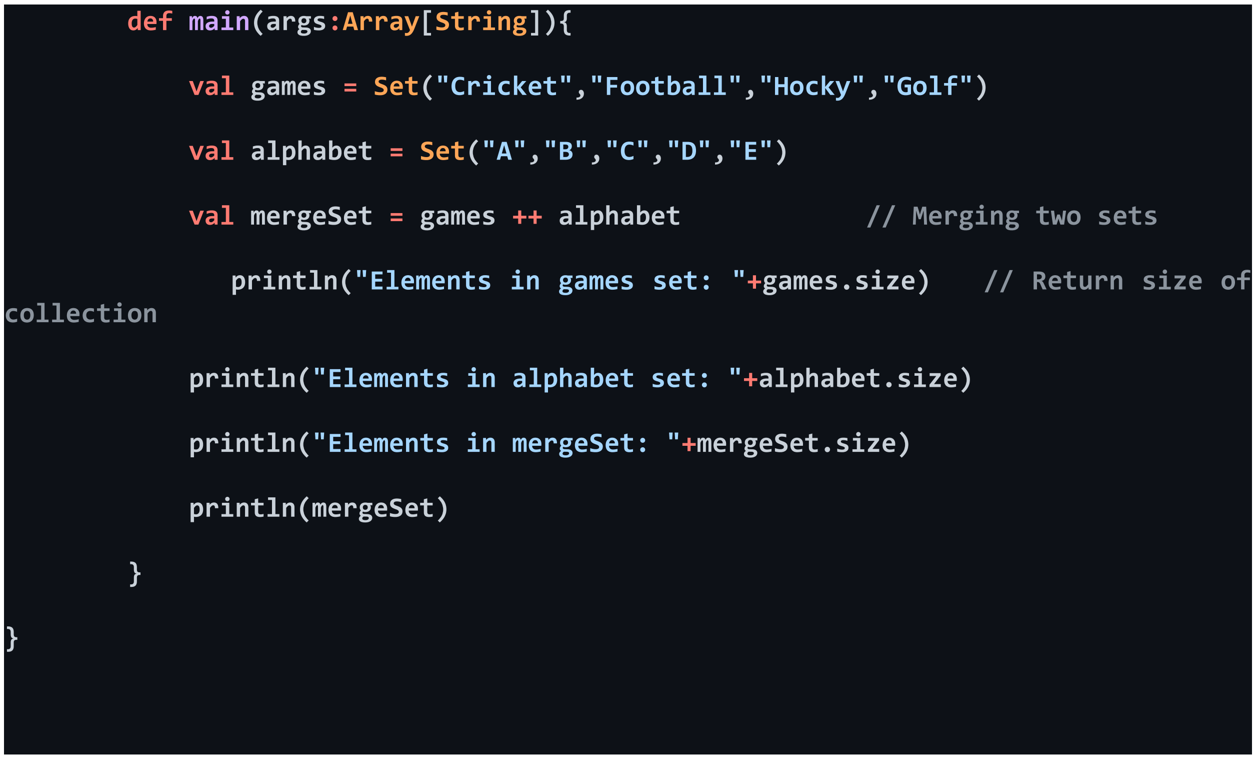


```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
list1 : List(1, 2, 3)
list2 : List(4, 5, 6)
Merging list1 and list2
Merged list : List(1, 2, 3, 4, 5, 6)

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```

3.

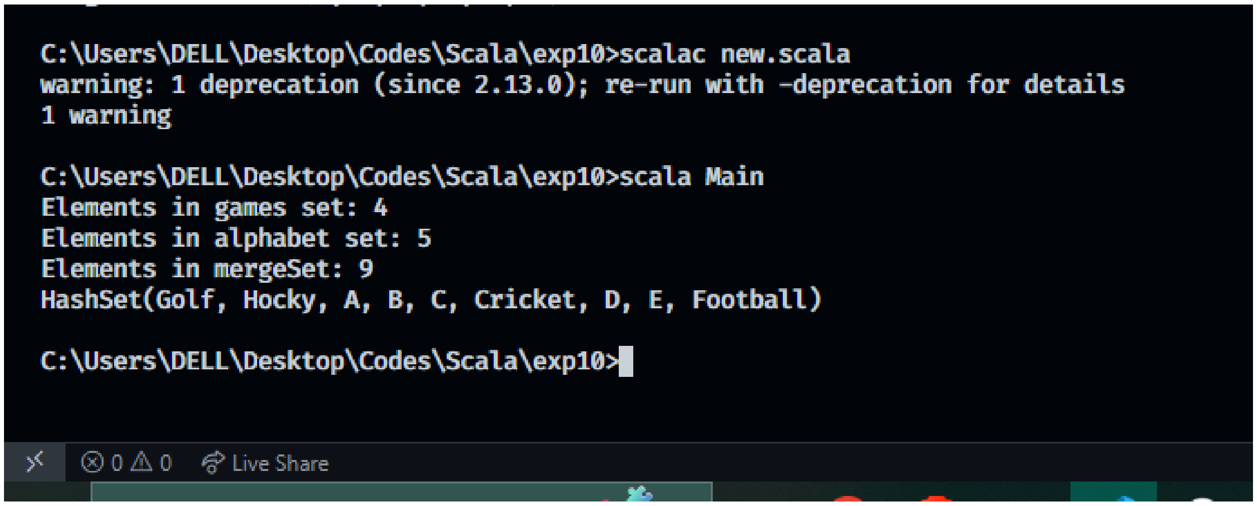
Code :

```
import scala.collection.immutable._

object Main{
```



```
        def main(args:Array[String]){

            val games = Set("Cricket","Football","Hocky","Golf")

            val alphabet = Set("A","B","C","D","E")

            val mergeSet = games ++ alphabet            // Merging two sets

               println("Elements in games set: "+games.size)    // Return size of
collection

            println("Elements in alphabet set: "+alphabet.size)

            println("Elements in mergeSet: "+mergeSet.size)

            println(mergeSet)

        }

}
```

Output:



```
C:\Users\DELL\Desktop\Codes\Scala\exp10>scalac new.scala
warning: 1 deprecation (since 2.13.0); re-run with -deprecation for details
1 warning

C:\Users\DELL\Desktop\Codes\Scala\exp10>scala Main
Elements in games set: 4
Elements in alphabet set: 5
Elements in mergeSet: 9
HashSet(Golf, Hocky, A, B, C, Cricket, D, E, Football)

C:\Users\DELL\Desktop\Codes\Scala\exp10>
```